Application of Differential Evolution algorithms to multi-objective optimization problems in mixed-oxide fuel assembly design

Alan Charles [1,*], Geoffrey Parks [1]

[1] University of Cambridge, Department of Engineering, Trumpington Street, Cambridge, CB2 1PZ, United Kingdom

* Corresponding author.
E-mail addresses: ajc289@cam.ac.uk (Alan Charles), gtp10@cam.ac.uk (Geoffrey Parks)

Abstract

Multi-objective optimization of nuclear engineering fuel assembly design problems is particularly difficult due to the highly non-linear interactions of a large number of possible variables. In addition, effective optimization algorithms are often highly problem-dependent and require extensive tuning, which reduces their applicability to the real world. To address this issue, Differential Evolution (DE) algorithms have been proposed as a new and effective method for heterogeneous fuel assembly optimization design problems. This paper presents the first complete study to investigate their applicability and performance. Firstly, two multi-objective DE algorithms have their performance compared against an Evolutionary Algorithm (EA) from the literature in optimizing a CORAIL mixed-oxide (MOX) fuel assembly for maximum plutonium content and minimum power peaking factor. Statistical analysis of the results shows the DE algorithms exhibit superior performance to the EA. The DE algorithms are then used to optimize a MOX fuel assembly with gadolinia poison, with results showing DE produces assembly designs comparable in performance to those in the literature. Finally, a sensitivity study is conducted on the control parameters of the better performing of the DE algorithms. Results indicate DE performance remains consistent for a wide range of values of both control parameters, suggesting the algorithm is able to perform effectively without requiring user expertise or effort to find the 'optimal' control parameter settings.

1. Introduction

Varying the properties of fuel pins on a pin-by-pin basis across a nuclear reactor fuel assembly, both axially and radially, can potentially provide benefits for various fuel and core performance and safety criteria. However, the large number of variables, their non-linear interaction and the number of possible combinations makes it extremely difficult to quantitatively define the trade-offs between the different performance criteria which are often in competition, such as k-effective and the Power Peaking Factor (PPF). By giving engineers a capability to rigorously and systematically explore the trade-offs involved in design, optimization of both existing and new designs becomes possible in an area where optimization is too difficult to achieve through conventional engineering judgement alone.

Optimization refers to the process of attempting to determine which combinations of variables within a system produce solutions which achieve the best performance or are closest to pre-defined performance objectives. In real-world engineering problems, much of the information about the system is incomplete, and this is particularly true of nuclear engineering design problems which often feature high dimensionality. Optimization methods have been providing effective solutions for core loading pattern design problems since the 1970s using a

variety of techniques from Linear Programming (Suzuki and Kiyose, 1971), to Simulated Annealing (Kropaczek and Turinsky, 1991), Genetic Algorithms (Parks, 1996), Particle Swarm Optimization (Khoshahval et al., 2011) and Tabu Search (Jagawa et al., 2001). This is the focus of the majority of nuclear engineering optimization research. However, optimization methods can also be applied to aspects of the design of the fuel assembly itself. Generally, this sort of optimization involves varying the uranium-235 enrichment in the assembly (Hirano at al., 1997), along with other influencing factors, such as the number of gadolinium pins (Yamate et al., 1997), in order to optimize some objectives, usually to minimize the PPF or to maximize the infinite multiplication factor (Castillo et al., 2011). Given the natural trade-offs entailed with competing objectives such as these, a multi-objective approach can prove to be highly effective, with some studies demonstrating results which surpass expert designs (Lattarulo et al., 2014; Charles and Parks, 2017).

Single-objective optimization methods are designed to improve one objective in isolation by changing other variables, whereas multi-objective methods attempt to improve two or more figures of merit simultaneously. The most common techniques for solving multi-objective problems are either to construct a single composite objective function using a weighted summation of the individual objectives, or to constrain all but one of the objectives and focus optimization upon the remaining objective (Sawaragi et al., 1985). An obvious problem with these approaches is that the constraint limits and weightings employed will almost certainly be based on the designer's judgement, and thus the approach (and the solution obtained) is subjective (Parks, 1996). To avoid reliance on this, one can use the concept of 'dominance' to find the area of global optimization (the trade-off surface or Pareto front).

Any solution on the Pareto front can be identified formally by the fact that it is not dominated by any other possible solution. One solution is said to be *dominated* by another, if the latter is at least as good on all counts (objectives) and better on at least one. This is illustrated in Fig. 1: solutions P2 and P3 both dominate P1. By evaluating and comparing all possible solutions, the set of optimal nondominated solutions (the Pareto front) can be found for the problem at hand, as shown in Fig. 2. Exhaustive enumeration of all possible solutions is usually not practicable for real-world problems, and thus the Pareto front found is an approximation of the true one – a good approximation if the multi-objective optimization algorithm used performs well.

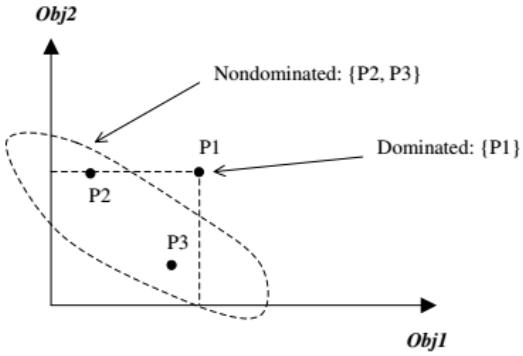

Fig. 1. Nondominated and dominated solutions in a two-objective (both to be minimized) optimization problem (Pereira, 2004).

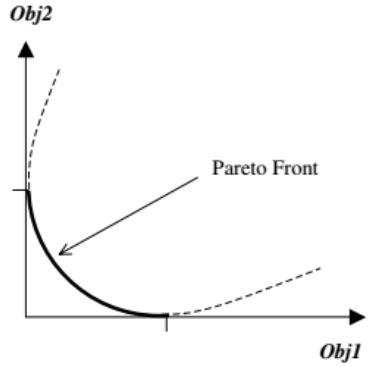

Fig. 2. An example of a Pareto front in a two-objective (both to be minimized) optimization problem (Pereira, 2004).

As successive iterations of the optimization algorithm are executed, improved Pareto fronts can be formed until a convergence or termination criterion is met. A class of algorithms known as 'multi-objective metaheuristics' have been shown to be highly effective when combined with suitable evaluation software, such as the reactor physics analysis package WIMS (Lindley et al., 2015). These are algorithms which learn information about the system they are investigating as they progress, and modify how they work in light of this experience in order to accelerate the search for optimal solutions.

The most well-known class of metaheuristics is Evolutionary Algorithms (EAs) (Coello Coello et al., 2007). These attempt to mimic the processes involved in evolution by natural selection – by combining superior solutions through a process known as 'crossover' or 'recombination' to create new 'offspring' or 'generations' of solutions, with random mutations attempting to ensure diversity within the 'population'. A downside of EAs is the many control parameters which govern the behaviour of the algorithm, making certain operations (such as crossover or mutation) more or less likely depending on the values of these parameters. When applied to a new problem for which little is known about the search space, where optimal solutions might lie, or how the design variables interact (as is most often the case with engineering problems), these algorithms may exhibit poor performance if a period of parameter 'tuning' is not performed first. This results in increased computational requirements and can potentially mislead the engineer about the nature of the search space. Tuning can be simplified significantly if the algorithm contains some form of adaptive parameter control, whereby the parameters are dynamically adjusted using feedback from the search process, enabling the algorithm to adjust itself as it searches, leading to faster and more reliable convergence.

In contrast to traditional EAs, Differential Evolution (DE) algorithms (Storn and Price, 1997) are a relatively newer type of EA that work in a similar fashion but feature key differences in the way the new population is generated. Instead of using a predefined probability density function, new solutions are generated through taking a scaled difference between two parents (known in DE as the mutation strategy) and adding it on to a third parent in a crossover process. This results in a selection process that is generally more stringent than that used in GAs (where inferior solutions have a probability of remaining in the population), because only solutions which are better than or at least as good as the current population are allowed to survive to the next generation. This gives DE algorithms a potentially faster convergence rate by being inherently greedier (which is useful for computationally expensive problems, such as those typically faced in nuclear engineering, as it reduces the number of evaluations necessary to reach a high quality solution). However, greedy algorithms typically have a higher risk of losing diversity in the population. Without diversity the algorithm can

prematurely converge on a solution which is a local optimum, rather than a global optimum, simply because it is unaware other possibilities exist (Zio and Viadana, 2011). DE algorithms have previously been successfully applied to nuclear reactor core design optimization problems (Sacco et al., 2009); however, they do not yet appear to have been applied to nuclear fuel assembly design optimization problems, thus making this investigation both novel and a useful step in examining DE's applicability to solving such problems.

This research investigates the performance of multi-objective DE algorithms on nuclear engineering fuel assembly design problems (compared to EAs) and the sensitivity of that performance to control parameter settings, in order to determine their overall suitability for optimization problems of this kind. For this work, new multi-objective forms of the DE algorithms JADE (Zhang and Sanderson, 2009) and μJADE (Brown et al., 2015) are developed.

JADE was developed to offer a potentially superior mutation strategy by combining a 'greedy' method with an external archive. The mutation strategy is one of the defining characteristics of a DE algorithm and choosing which parents to perform a weighted difference on affects how wide or narrow the search potential is. Less greedy strategies involve choosing random parents, whereas in more greedy strategies parents are only chosen from the 'best' or 'set of best' solutions. Including an archive allows the algorithm to take information not only from the 'best' solutions (which in multimodal problems are harder to define), but also from other previously found solutions, which enables the algorithm to maintain a diverse set of solutions to mitigate against the risk of premature convergence. This risk is then further reduced by using adaptive parameter control. Both JADE and μJADE feature adaptation of the control parameters and have just two control parameters set by the user: the rate of parameter adaptation, and the elitism/greediness of the selection step. This contrasts with other algorithms without adaptive parameter control, which can feature 8–10 separate control parameters which must be set by the user. As pointed out by Zhang and Sanderson (2009), adaptive algorithm control improves algorithm robustness and reduces the need to tune control parameters to individual problems. JADE has been shown in the literature to exhibit superior performance over classic DE algorithms (Zhang and Sanderson, 2009). μJADE was chosen as it has also been shown to be effective when working on multimodal problems (Brown et al., 2015), but uses a significantly smaller population size, which is of great benefit when analyzing real-world problems where evaluating the population is far more computationally expensive.

This paper applies DE algorithms to three problems. The first problem investigated concerns optimization of a so-called 'CORAIL' assembly (Youinou et al., 2001) containing both low-enriched uranium (LEU) and plutonium mixed-oxide (MOX) pins, with the objectives of minimizing the PPF and maximizing plutonium content. This problem serves to compare the performance of the DE algorithms and a representative EA. The second problem involves optimization of a MOX assembly which includes gadolinium burnable absorber pins, investigating the performance of the DE algorithms on a more complex problem. Finally, the sensitivity of the better performing DE algorithm is investigated. Low sensitivity indicates that the algorithm performs robustly and does not require excessive parameter tuning before it can be run on a new problem.

2. Optimization algorithms

DE algorithms search using a 'population' of 'solutions', where in this work each 'solution' is a possible fuel assembly design generated by the algorithm. In each 'generation', solutions are evaluated to determine which designs are better at achieving a set of given objectives. The next generation of the population is then determined by following the

principle of natural selection, whereby 'good' solutions have a greater influence on subsequent generations of solutions than 'poor' solutions. Solutions are evaluated to determine their objective values.

2.1. JADE, MOJADE and MOμJADE

The DE algorithm JADE was chosen from the literature as the basis from which to develop the multi-objective forms. JADE has already proven to be highly competitive on optimization problems with high dimensionality (Zhang and Sanderson, 2009). It features a mutation strategy called "DE/current-to-pbest/1", shown below in Eq. (1), which creates a mutated solution:

$$v_{i,g} = x_{i,g} + F_i \times \left(x^p_{best,g} - x_{i,g}\right) + F_i \times \left(x_{r1,g} - \bar{x}_{r2,g}\right) \tag{1}$$

Here $v_{i,g}$ is the mutated solution, $x_{i,g}$ is a current member of the population, $x_{r1,g}$ is a randomly chosen member of the population, $\bar{x}_{r2,g}$ is a randomly chosen solution from an archive of 'worst' solutions, $x^p_{best,g}$ is a solution from an archive of 'best' solutions, and F_i is a weighting factor, determined from the mutation rate. The mutation rate and the crossover rate are changed according to how successful the algorithm has previously been at creating superior solutions and are regulated (after each generation) by the control parameter c (the rate of parameter adaptation). The size of the archive of 'best' solutions is determined by control parameter p (the elitism of the algorithm). p and c are the only user-specified control parameters in JADE, making it inherently easier (compared to other EAs) to tune performance.

μJADE uses a slight variant to this, called "DE/current-by-rand-to-pbest/1" (Brown et al., 2015), shown below in Eq. (2):

$$v_i = x_i + F_i \times \left(x^p_{best} - x_a\right) + F_i \times \left(x_b - \tilde{x}_c\right) \tag{2}$$

Common terms with Eq. (1) have the same meaning here, while x_a and x_b are randomly chosen members of the population that are not x_i, and $\tilde{x}_c$ is a randomly chosen member of the archive of 'worst' solutions. This feature keeps the strategy exploratory during early stages of optimization, and, as the population converges, x_i becomes closer to x_a and the strategy becomes closer to the "DE/current-to-pbest/1" strategy featured in JADE, accelerating convergence. Once again, the mutation rate and crossover rate, along with the greediness, are regulated by the same control parameters as in JADE. Both JADE and μJADE feature a binomial crossover method, which is the standard method for DE, and allows for any combination of mutated and non-mutated components. Alternatives include the exponential crossover method, which crosses over a number of consecutive components, but this is generally inferior to binomial crossover (Zio and Viadana, 2011).

Multi-Objective JADE (MOJADE) and μJADE (MOμJADE) were created using C++ and are based on the JADE and μJADE algorithms, as described by the originators in (Zhang and Sanderson, 2009) and (Brown et al., 2015), respectively, with the following modifications implemented to allow them to operate in a multi-objective environment. First, selection and ranking are no longer done based on one objective – this was changed to use the concept of dominance to determine the Pareto front. Therefore, the 'best' solutions are now a list of nondominated solutions, which represent the current Pareto front, the trade-off in the objectives of the solutions found thus far. Secondly, archiving was changed such that it now

takes dominated solutions from the population. An additional archive was added to accept new solutions that are Pareto-equivalent to (i.e. neither dominated nor dominating) the existing population. Therefore, $\bar{x}_{r2,g}$ for MOJADE and $\tilde{x}_c$ for MOμJADE can be any solution from the archives of dominated and Pareto-equivalent solutions. MOJADE and MOμJADE were initially tested on the ZDT-1 problem (Zitzler et al., 2003) and their results were compared to those obtained using the NSGA-II algorithm (Deb et al., 2002; Knowles et al., 2006), using a test case of 41 dimensions. The test confirmed that both DE algorithms were able to find the Pareto front with performance comparable to the NSGA-II algorithm, and confirmed that they can successfully operate in a multi-objective environment.

Pseudocode for MOJADE and MOμJADE can be found in Appendix A. Control parameters used for MOJADE and MOμJADE in this work are given in Table 1.

Table 1
MOJADE and MOμJADE control parameters.

Parameter	MOJADE	MOμJADE
Rate of parameter adaptation c	0.1	0.05
Greediness of selection strategy p	0.05	3 / population
Population size	32	8
Generations	50	200

2.2. Multi-Objective Alliance Algorithm

In order to assess the performance of these newly created DE algorithms, they are compared to an algorithm from the literature that has previously demonstrated effectiveness in optimizing nuclear fuel assembly design problems – the Multi-Objective Alliance Algorithm (MOAA) (Lattarulo and Parks, 2012).

The MOAA is a metaheuristic optimization algorithm inspired by the metaphorical idea of a number of tribes struggling to conquer an environment offering resources that enable them to survive. The tribes are characterized by two features: the skills and resources necessary for survival. Tribes try to improve skills by forming alliances, which are also characterized by the skills and resources needed, but these now depend on the tribes within the alliance. The two main search elements of the algorithm are the formation of alliances and the creation of new tribes. One MOAA cycle ends when the strongest possible alliances of existing tribes have been created. The algorithm then begins a new cycle starting with new tribes whose creation is influenced by the previous strongest alliances.

Tribes are initially created randomly, but, once a Pareto front has been established, they become either copies of Pareto front solutions or are modified from the Pareto front using a normal distribution. This distribution has an adaptive standard deviation to increase diversity initially and then speed up convergence towards the end of the optimization. The algorithm also analyses the distance between solutions on the Pareto front to determine which solutions to remember. This feature also functions adaptively: as the algorithm converges and the average gap between solutions becomes smaller, dominated solutions near areas of the Pareto front that have larger gaps are preserved in an archive to encourage the finding of a non-dominated solution in that area in the future. These features are all governed by control parameters, and the (default) values of the MOAA control parameters used in this work are given in Table 2.

Further details concerning the application of the MOAA to nuclear fuel assembly design can be found in (Lattarulo et al., 2014). In that case study, the MOAA found solutions superior to previous ‘expert designs’ and out-performed other EAs.

Table 2
MOAA control parameters.

Parameter	Value
Number of tribes	6
Probability 1 for the creation of tribes	0.5
Probability 2 for the creation of tribes	0.2
Initial standard deviation	0.3
Final standard deviation	0.01
Probability 3 for the creation of alliances	2 / variables
Alliance standard deviation	0.1
Total number of Pareto-optimal solutions	100
Factor for evaluation neighbourhood	10

3. Test problems

3.1. Problem 1

The first problem investigated was originally presented in (Lattarulo et al., 2014) and DE results were first presented in (Charles and Parks, 2017). The task is to optimize a two-dimensional nuclear fuel 'CORAIL' type assembly containing two types of fuel pin, LEU and uranium-plutonium MOX (see Fig. 3).

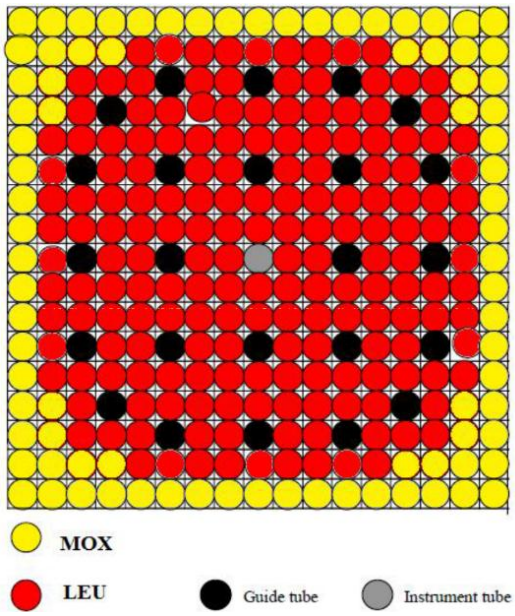

Fig. 3. CORAIL assembly with LEU pins surrounded by MOX pins at the periphery (Lattarulo et al., 2014).

The presence of both Pu and LEU results in a wider neutron energy spectrum inside the reactor during operation, creating uneven reaction rates, variations in the radial neutron flux and power distribution, and can potentially result in fuel temperature problems. By optimizing the distribution of pins inside the assembly this imbalance can be minimized. Optimization can be carried out by changing both MOX pin positions and the concentration of plutonium within the MOX pins, as Lattarulo et al. (2014) demonstrated, increasing the overall Pu content above that of the standard CORAIL expert design. For reasons of safety, at least half the total number of pins should always be LEU only and the %Pu within the MOX pins can

be no more than 20%. The plutonium composition was assumed to be reactor grade, and is detailed in Table 3.

Table 3
Plutonium isotopic composition (%) used for Problem 1.

^{238}Pu	^{239}Pu	^{240}Pu	^{241}Pu	^{242}Pu	^{241}Am
3.90	40.57	30.08	12.32	11.89	1.24

LEU enrichment is kept fixed at 5% ^{235}U. The geometry was fixed to be that of a standard CORAIL assembly containing 264 fuel pins. Using octant symmetry this can be simplified to give 39 unique fuel pin positions. Pin types 1, 2 and 3 refer to MOX type 1, MOX type 2 and LEU, respectively. N_1, N_2 and N_3 are therefore the quantities of each pin type, with the sum total being equal to the number of pins in the assembly octant ($N_1 + N_2 + N_3 = 39$). Two MOX pin types are allowed with different %Pu amounts (W_1, W_2). The constraints are $N_3 \geq$ 16.5 (264/8), which represents a lower limit for the number of LEU pins in the octant due to safety reasons, and $0 \leq W_1, W_2 \leq 20$, which represent the range of possible %Pu values. The total plutonium content in the assembly is given by $\mathrm{MOXT} = W_1 \cdot N_1 + W_2 \cdot N_2$. Pins along the lines of octant symmetry within the assembly are weighted by 0.5 when calculating the value of MOXT, to avoid double counting of Pu. The objectives to be minimized are PPF at beginning of life (BoL) and –MOXT. PPF values are obtained using the reactor physics code WIMS10a (Lindley et al., 2015) to solve the neutron transport equation, using the method of characteristics, to calculate pin power and hence the PPF. To calculate the PPF, WIMS fixes the mean pin power.

MOJADE, MOμJADE and MOAA were each run 30 times, with a unique random seed each time. Each individual run had a limit of 1600 solution evaluations, which allowed for 50 generations of MOJADE using a population of 32, and 200 generations of MOμJADE using a population of 8. Algorithms were run on the 'Ray' computer cluster used by the University of Cambridge's Department of Engineering Nuclear Group, with specifications shown in Table 4.

Table 4
Ray computer cluster specifications.

Processor	Intel Xeon Processor E5-2650 (2.6 GHz, 20 MB cache)
Threads	16
RAM	64 GB DDR3

3.2. Problem 2

The second test problem was chosen to investigate the effectiveness of DE on a more complex problem without performing any control parameter tuning. This problem concerns the optimization of MOX fuel assemblies containing gadolinia (Gd_2O_3) pins, e.g. Japanese MOX assemblies (Yamate et al., 1997). The use of gadolinia pins in these assemblies potentially reduces the need to use burnable poison rods (BPRs) in the guide tubes, normally employed to compensate for higher PPF values caused by higher levels of Pu content compared to other designs. By optimizing the design using gadolinia pins, the PPF can be reduced without using BPRs and can even allow for increased Pu content in the assembly.

In (Yamate et al., 1997), one assembly was optimized for minimum PPF over the life of the assembly, using a fixed %Gd content, fixed pin types and changing %Pu contents for two types of U-Pu MOX pin. To match the original paper, a slightly different plutonium composition was used to mimic Japanese-style MOX pins (see Table 5).

Table 5
Plutonium isotopic composition (%) used for Problem 2.

^{238}Pu	^{239}Pu	^{240}Pu	^{241}Pu	^{242}Pu	^{241}Am
1.90	57.50	23.30	10.00	5.40	1.90

Using multi-objective optimization algorithms, it is possible to further explore the search space for this problem, with the objectives once again of maximizing plutonium content and minimizing the PPF at the assembly BoL. The design variables were changed to include all five originally proposed assembly layouts, allowing %Gd and %Pu to change, and allowing all non-Gd pins to be of either type of Pu MOX pin. The different assembly layouts used are shown in Fig. 4.

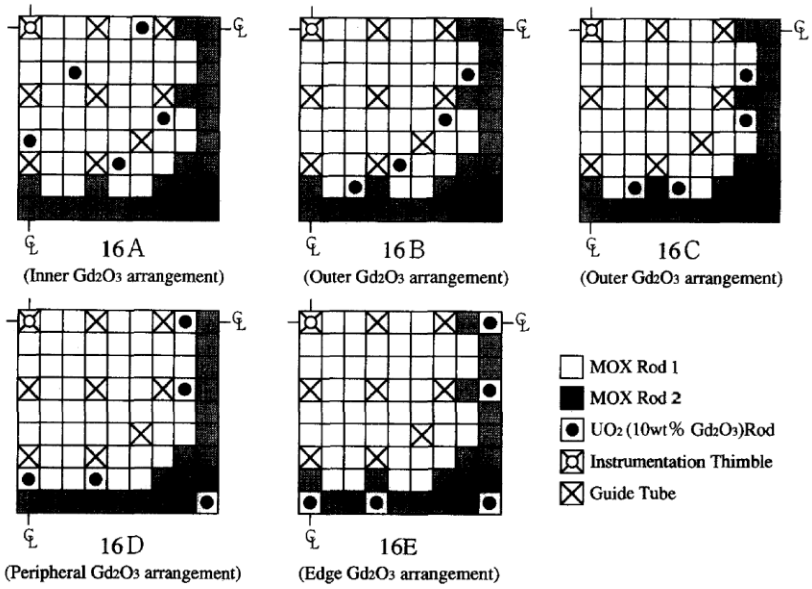

Fig. 4. Japanese U-Pu MOX ¼ assembly layouts used in Problem 2 (taken from (Yamate et al., 1997)).

Similar to the previous problem, the assembly contains 264 fuel pins and has 39 unique fuel pin positions, for octant symmetry. Pins are labelled as fuel types 1, 2 and 3 (MOX type 1, MOX type 2, and gadolinia, respectively). Total numbers of each pin type are given by N_1, N_2 and N_3, with $N_1 + N_2 + N_3 = 39$. The quantity and positions of the gadolinia pins are dependent on which assembly layout is chosen, from the five possibilities (shown in Fig. 4), with some pins weighted by 0.5 due to octant symmetry in the assembly. The two MOX pin types can be placed anywhere in the assembly except at guide tube or gadolinia pin locations. The two %Pu weights are W_1 and W_2, and one concentration of gadolinia is allowed (W_G). Constraints of $0 \leq W_1, W_2 \leq 20$, and $0 \leq W_G \leq 10$ were used. The total Pu content is again $\mathrm{MOXT} = W_1 \cdot N_1 + W_2 \cdot N_2$.

Both MOJADE and MOμJADE were run on this problem and their performance compared. Following this, depletion of a solution on the elbow of the found Pareto front solution was performed to see how the PPF changed over the life, with results compared to those in (Yamate et al., 1997). Both algorithms were used with the same control parameter values as used for the first problem to see how well the algorithms performed without custom tuning of parameters. Again, the reactor physics code WIMS10a (Lindley et al., 2015) was used to calculate PPF values and to perform depletion calculations. MOJADE was run with a

population of 32 for 50 generations. MOμJADE was run with a population of 8 for 200 generations, giving both algorithms a total of 1600 function evaluations in each run. Both algorithms were run 20 separate times. Algorithms were run on the 'Lux' computer cluster used by the University of Cambridge's Department of Engineering Nuclear Group, with specifications shown in Table 6.

Table 6
Lux computer cluster specifications.

Processor	Intel Xeon Processor E5-2690 (3.5 GHz, 35 MB cache)
Threads	28
RAM	128 GB DDR3

3.3. Sensitivity analysis

The sensitivity of DE to the values of its control parameters was measured using the same optimization problem described in Sect. 3.2. Only MOJADE was investigated in this case, since it was shown to perform better than MOμJADE on Problems 1 and 2, and is arguably more suited to nuclear engineering problems where parallelization of the evaluation step offers a significant execution time advantage.

Constraints were kept the same, and the optimization objectives were again to maximize the Pu content of the assembly and minimize the BoL PPF. It was decided to confine the study to looking at assembly performance at BoL to reduce the computational cost of the investigation, as including burnup calculations in the evaluation step increases the computational load significantly. It was judged that the BoL design problem was sufficiently complex to provide a good test of the performance sensitivity to the rate of adaptation c and the greediness of the algorithm p.

MOJADE was run with a population of 32 for 40 generations. The focus of this study was on the impact on performance of the algorithm's degree of elitism and self-adaptive nature, as the trade-off between increased population size providing more diversity and greater search space coverage versus computational load is already well established. Table 7 shows the control parameter ranges tested along with their default values. Work by the original authors of JADE suggests that the rate of parameter adaptation c works well with values in the range 0.05–0.2, and the greediness p works well between 5 and 20% (i.e. the 'best' results are chosen from between 5 and 20% of the current population) (Zhang and Sanderson, 2009). Both very high and very low values of p and c were investigated to determine the effect these control parameters have on the algorithm's performance. Each test was run 10 times (varying only the random number generator seed used in each run) to obtain a suitable statistically significant set of results. Runs were executed on the 'Lux' computer cluster.

Table 7
MOJADE control parameter values used in sensitivity analysis tests.

Test number	Greediness of selection strategy, *p*	Rate of parameter adaptation, *c*
Default Values	0.05	0.1
1	0.05	0.0
2	0.05	0.025
3	0.05	0.25
4	0.05	0.75
5	0.05	1.0
6	0.01	0.1
7	0.25	0.1
8	0.75	0.1
9	1.0	0.1

4. Results and discussion

4.1. Problem 1

The output of each run was the final Pareto front found by the algorithm. The results were analyzed by comparing these Pareto fronts. Analysis as presented in (Charles and Parks, 2017) involved using two separate indicators to determine the relative performance of each algorithm. Firstly, the epsilon indicator (Zitzler et al., 2003) represents the minimum translational distance necessary to move all points on a given Pareto front to weakly dominate a reference set (a combined Pareto front formed from all solutions from all algorithms representing the most optimal set of solutions). Secondly, the hypervolume indicator (Knowles et al., 2006) calculates the difference between the hypervolume of the dominated objective space formed from the Pareto front of one particular algorithm and the hypervolume of the objective space dominated by the reference set, using the least-optimal solution found as a reference point for the calculation of the hypervolume. In both cases smaller values indicate better performance. These same indicators will also be used later to analyze the results of the sensitivity study. To determine the statistical significance of the performance indicator values, the Kruskal-Wallis test was used (Kruskal and Wallis, 1952). For this work, the Kruskal-Wallis test results represent the probability that the given indicator values are not a true representation of the algorithm's relative performance against another, and are instead the result of random chance.

Results are plotted in PPF against (–MOXT) space. More negative values of –MOXT indicate a higher amount of plutonium in the assembly. Both objectives are to be minimized; therefore the bottom-left corner represents an ideal solution. Fig. 5 shows the results of every generated Pareto front for each algorithm. Fig. 6 shows these results filtered to show the overall best Pareto front for each algorithm. A line depicting the overall Pareto front formed from all the algorithms together is added for reference.

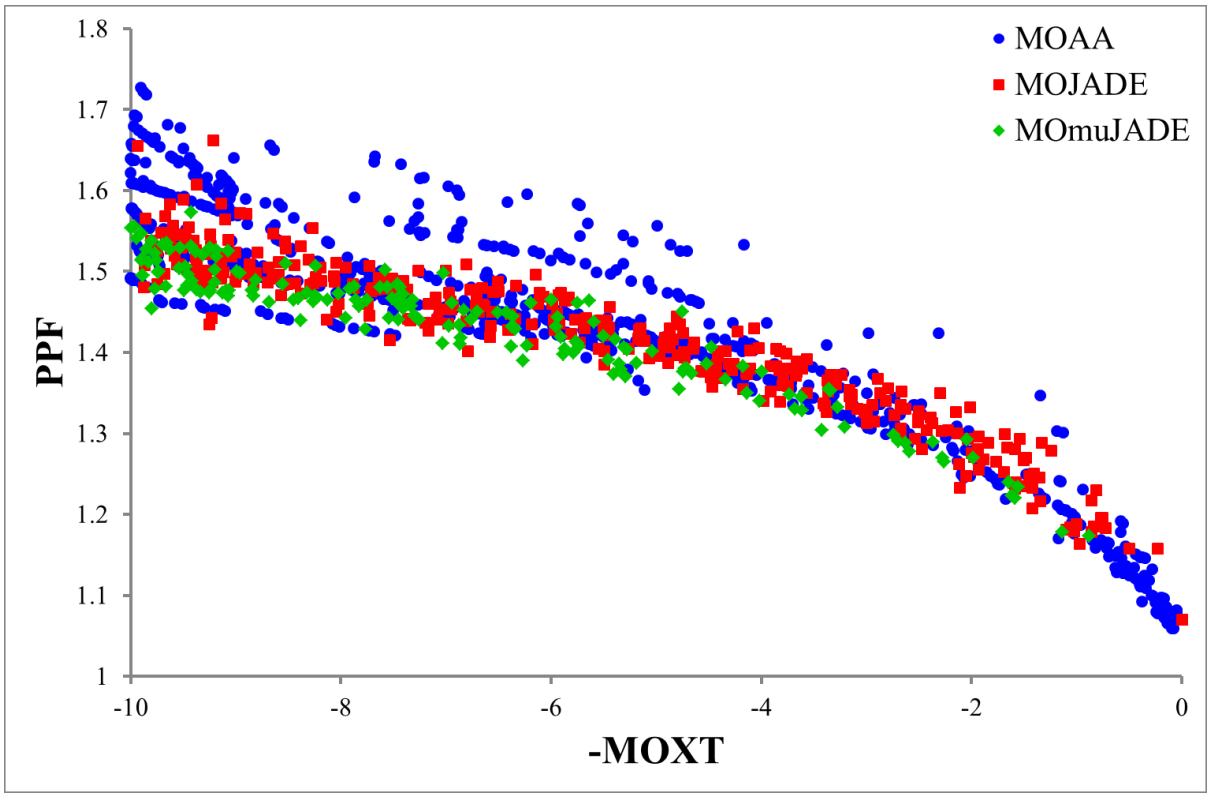

Fig. 5. Results of MOAA, MOJADE and MOμJADE optimization of MOX fuel assemblies in Problem 1, adapted from (Charles and Parks, 2017).

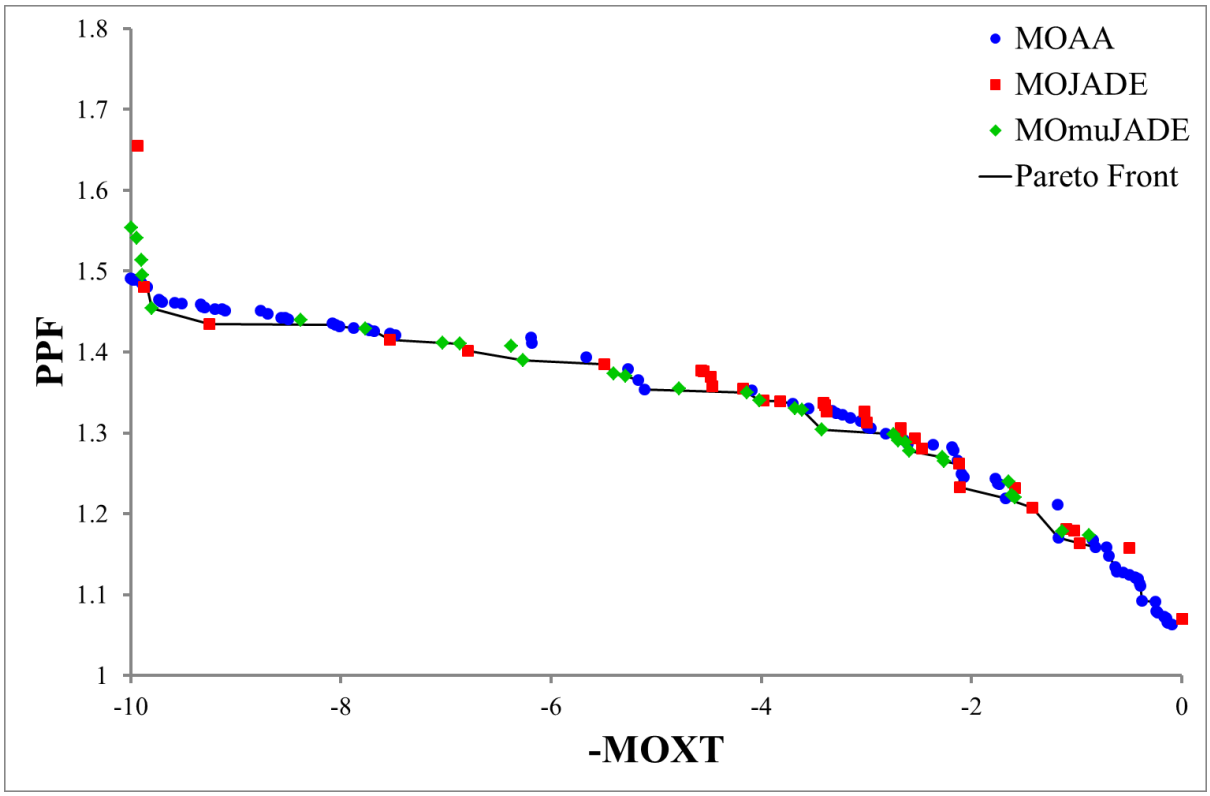

Fig. 6. Comparison of nondominated solutions found using the MOAA, MOJADE and MOμJADE algorithms to optimize MOX fuel assemblies in Problem 1, adapted from (Charles and Parks, 2017).

Figs. 5 and 6 demonstrate that MOJADE and MOμJADE perform comparably to MOAA, significantly contributing to the overall Pareto front, as highlighted in Fig. 6. The solutions found by MOAA appear to exhibit some degree of clustering in the Pareto front, with the gaps populated by MOJADE and MOμJADE solutions. MOAA tends to converge on a single MOX-LEU pin pattern during the course of a run, and thus the output from that run will typically be nondominated solutions which show the effect of increasing or decreasing the values of W_1 and/or W_2 within the same pin pattern. This results in a number of solutions that have very similar values for MOXT and PPF. In contrast, both MOJADE and MOμJADE do not necessarily converge on a single pin pattern in any given run, and thus arguably better explore the search space of different pin arrangements.

The means and standard deviations of the hypervolume and epsilon indicators, along with their corresponding p-values from the Kruskal-Wallis test, are given in Tables 8 and 9, respectively.

Table 8
Hypervolume and epsilon indicator values in Problem 1, adapted from (Charles and Parks, 2017).

Algorithm	Hypervolume Indicator		Epsilon Indicator	
	Mean	Standard Deviation	Mean	Standard Deviation
MOAA	1.6664	0.5169	0.3897	0.1478
MOJADE	0.7672	0.1047	0.3941	0.1204
MOμJADE	1.1267	0.7723	0.3320	0.1081

Table 9
Kruskal-Wallis test results in Problem 1, adapted from (Charles and Parks, 2017).

Algorithms	Hypervolume Indicator	Epsilon Indicator
MOJADE vs MOAA	3.879E-11	9.528E-01
MOμJADE vs MOAA	8.513E-07	7.363E-02
MOμJADE vs MOJADE	9.497E-05	5.650E-02

The hypervolume indicator results in Table 8 show that MOJADE is most consistent at producing results which dominate the entirety of the known search space, followed by MOμJADE. Results for the epsilon indicator, however, suggest that MOμJADE solutions are more likely to be closer to the 'true' Pareto front, but do not give as much information as to the exact nature of the Pareto front (MOμJADE search being limited by a smaller population size which leads to worse hypervolume indicator values).

Table 9 gives the p-value results of the Kruskal-Wallis test for the hypervolume and epsilon indicators for both DE algorithms versus MOAA, as well as against each other. Values lower than 0.05 indicate statistically significant (at the 5% level) results. The results indicate that MOJADE and MOμJADE yield superior hypervolume performance compared to MOAA due to the methodological differences in the algorithms. However, differences in epsilon indicator performance are not shown to be statistically significant. Finally, MOJADE shows superior hypervolume performance to MOμJADE, again due to methodological differences. In a given run for a fixed number of solution evaluations, the larger population of MOJADE is able to better cover the search space (and thus the Pareto front) compared to the small population of MOμJADE. There is some evidence that MOμJADE may be able to converge quicker than MOJADE and thus require fewer evaluations, which may offset the lack of inherent parallelization currently present in MOμJADE.

4.2. Problem 2

Fig. 7 shows the results given by MOJADE and MOμJADE, the Pareto front, and the solution chosen for depletion.

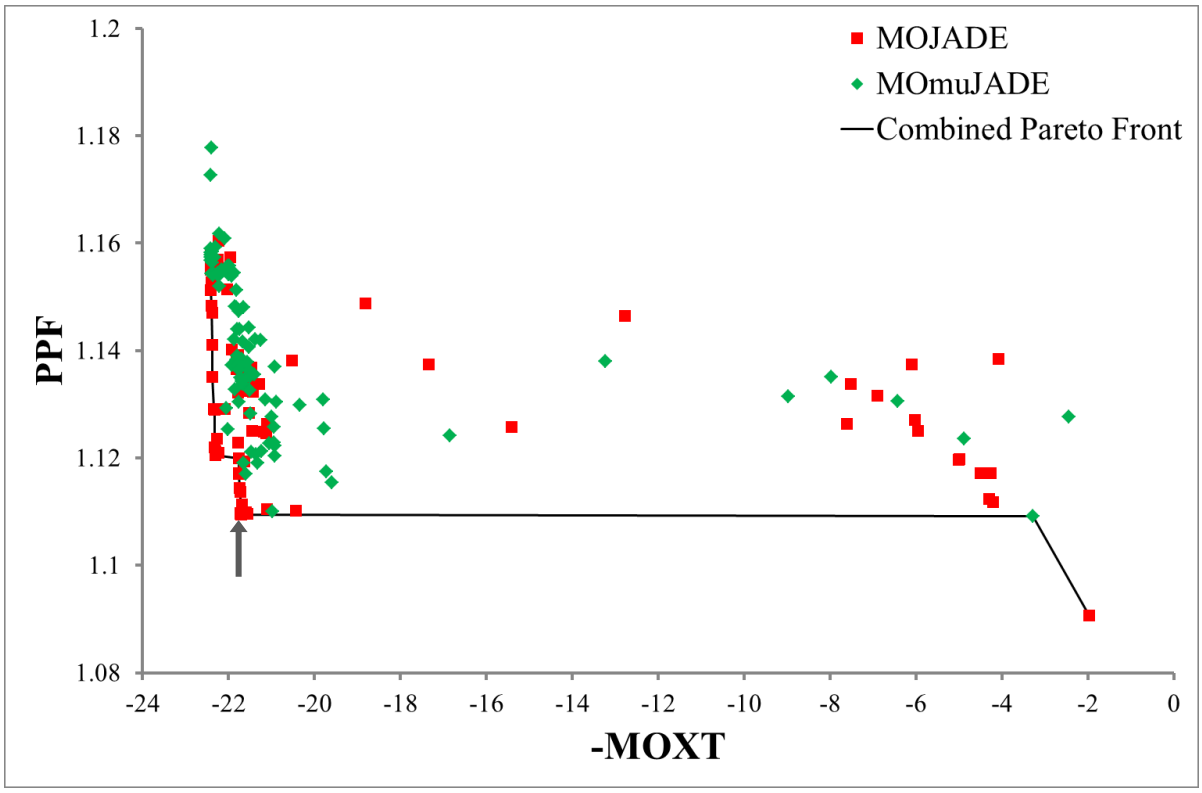

Fig. 7. Results of DE optimization of MOX fuel assemblies with gadolinia pins (Problem 2). The arrow indicates the solution chosen for the burnup study.

Fig. 7 indicates that both MOJADE and MOμJADE tend to converge on solutions containing high amounts of Pu, and the Pareto front for solutions with less negative values of –MOXT is poorly populated. It was originally thought that this may be due to some form of premature convergence causing a loss of diversity in the population around a local optimum of solutions containing high amounts of Pu. The crossover and mutation rates are self-adapting control parameters, which are, in turn, affected by the greediness p and the rate of parameter adaption c, as specified in Table 1. To test this hypothesis, a modified form of the problem was run with MOJADE, with the amount of Pu constrained such that solutions would only be permitted if the value of –MOXT was between –11 and –5. Fig. 8 shows that constraining the problem in this way results in a Pareto front that is dominated by the original results, and the solution clustering is a feature of the problem, not the algorithm.

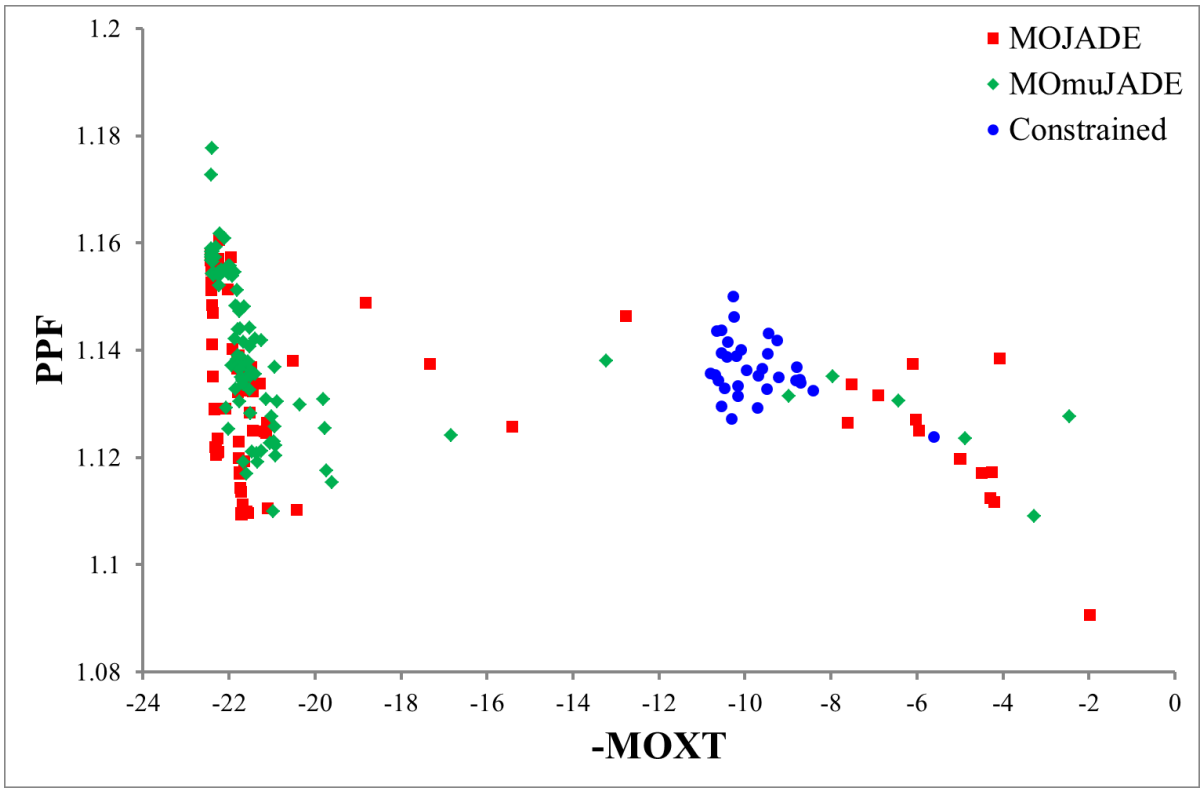

Fig. 8. The effect of constraining the %Pu within the MOX pins (Problem 2).

Figs. 7 and 8 only show PPF versus the total Pu content of the assembly, as these are the two objectives optimized; they do not show the amount of gadolinium contained within the gadolinia pins. For a given assembly layout and Pu content, changing the concentration of gadolinium will not only change the flux in nearby pins, but it will also cause the energy spectrum of the assembly to shift, dependent on the absorption cross-section of the gadolinia pins. Therefore increasing the gadolinium concentration may shift the spectrum in such a way as to cause the gadolinia to be less effective as an absorber, and thus potentially increase the PPF value. This highlights the complex and interrelated nature of the objectives when optimizing the design of a nuclear fuel assembly.

The Pareto front in Fig. 7 is almost entirely populated by MOJADE solutions. These results suggest that MOJADE performs better than MOμJADE on these types of problems, which supports the finding from Problem 1.

The solution found at the elbow of the Pareto front (shown by the arrow in Fig. 7) was depleted to 15 GWd/t. The evolution of the PPF against burnup for this MOJADE-generated assembly design can be seen in Fig. 9 as the blue line, overlaid on the original results of Yamate et al. (1997).

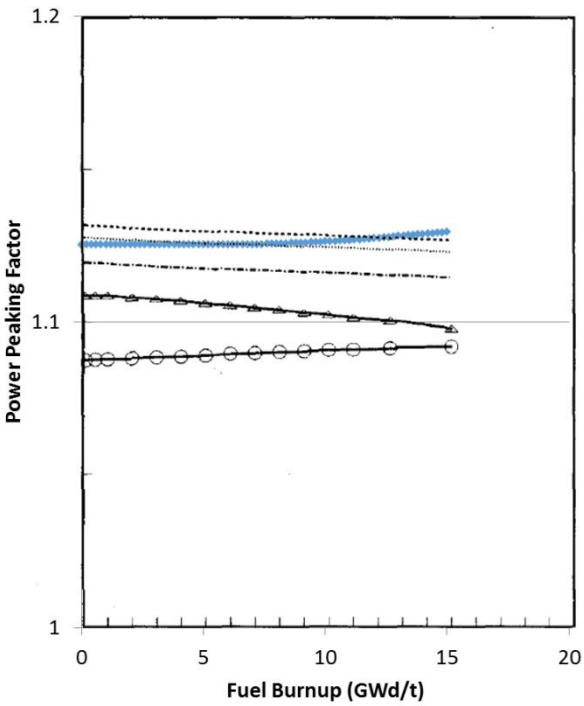

Fig. 9. PPF progression with burnup for gadolinia-MOX fuel assemblies with MOJADE-generated assembly design shown in blue, compared to other designs evaluated by Yamate et al. (1997), adapted from (Yamate et al., 1997).

The original paper (Yamate et al., 1997) investigated assemblies which ranged from 5.7% to 6.4% average Pu-pin wt% content. The chosen MOJADE solution had an average Pu-pin content of 19.5%. This test thus shows that DE algorithms are able to find designs that contain more Pu and keep internal PPF performance over one cycle comparable to that of assemblies with much lower Pu contents. Fig. 10 compares the assembly layouts of the depleted MOJADE solution and an example 'expert' assembly design from the (Yamate et al., 1997) study. The less conventional MOJADE design outperforms the 'expert' design, illustrating the solution space searching capability of a stochastic optimization algorithm.

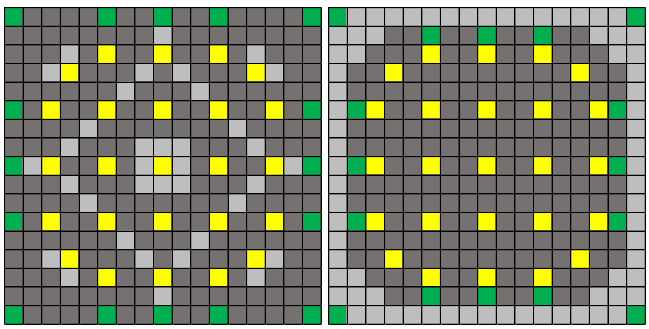

Fig. 10. U-Pu MOX assembly layouts with gadolinia poison rods produced using MOJADE (left) and from the literature (Yamate et al., 1997) (right). Light grey and dark grey indicate MOX pins (dark grey have higher %Pu contents), green indicates a poison rod, and yellow indicates guide tubes.

4.3. Sensitivity analysis

Figs. 11 and 12 show the Pareto fronts of each test using MOJADE with different control parameters in plots of PPF against –MOXT. Fig. 11 compares the Pareto fronts from each test run with the default parameter values and each test which changed the parameter adaptation rate *c*. Fig. 12 compares results with default parameter values to tests which changed the greediness of the selection strategy *p*.

As seen in Sect. 4.2, there is a high degree of result clustering, with few MOJADE solutions with –MOXT values below –15, for the reasons explained above. Fig. 11 indicates that larger rates of parameter adaptation may reduce MOJADE's ability to converge, whereas Fig. 12 suggests that varying the greediness parameter does not appear to have a large effect on the algorithm's performance for this problem.

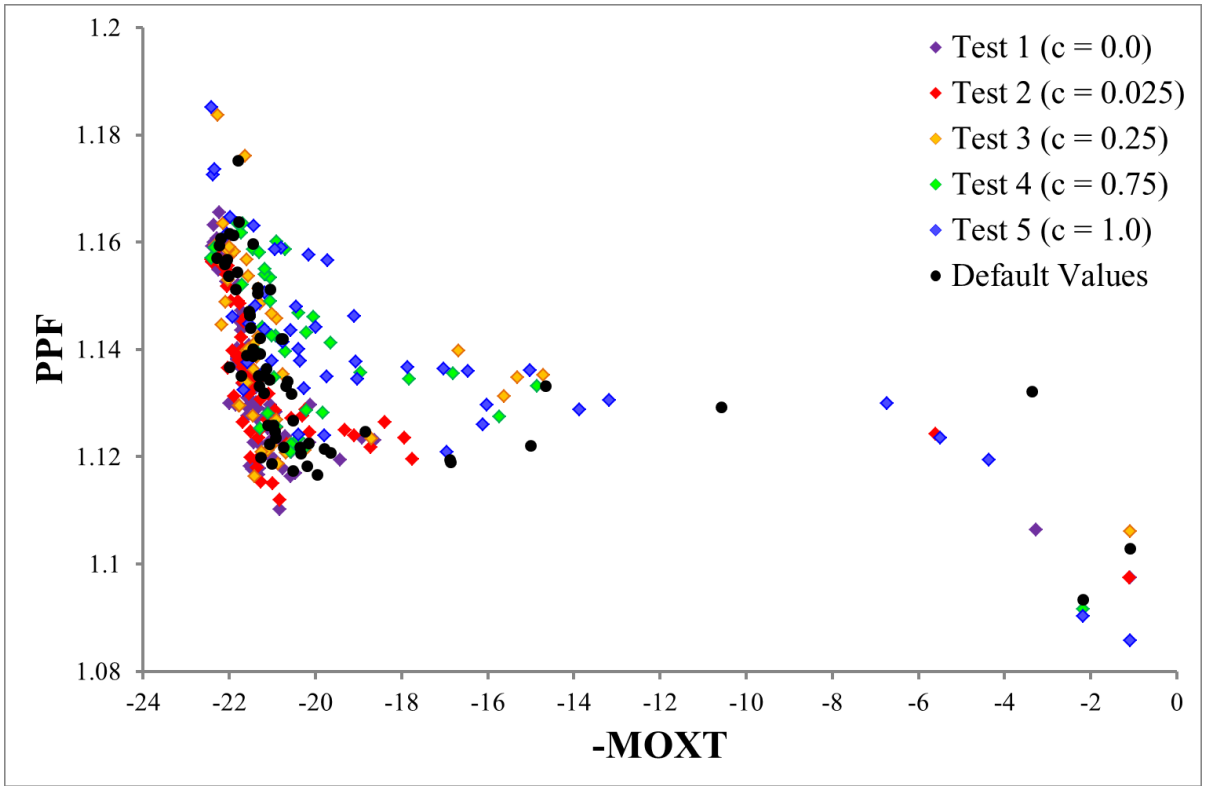

Fig. 11. Pareto front results for the parameter adaptation rate (*c*) sensitivity study.

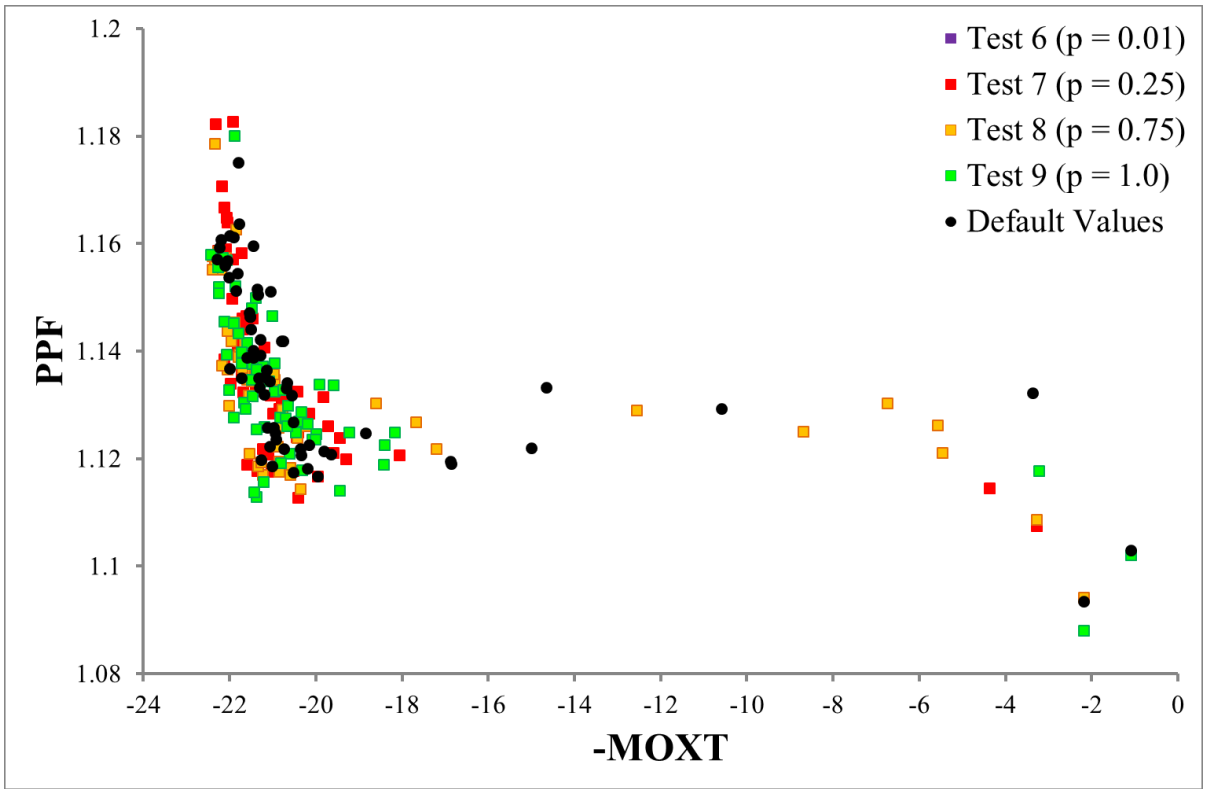

Fig. 12. Pareto front results for the greediness parameter (*p*) sensitivity study.

4.4. Statistical analysis of sensitivity study results

Once again, the hypervolume and epsilon indicator values were used to quantify performance. The progression of the hypervolume indicator value was monitored to confirm that the relative performance of the algorithm with different control parameter values had stabilized within the 40 generations allowed and would not significantly change were it run for more generations. This progression over the 40 generations (averaged over the 10 runs) can be seen in Fig. 13.

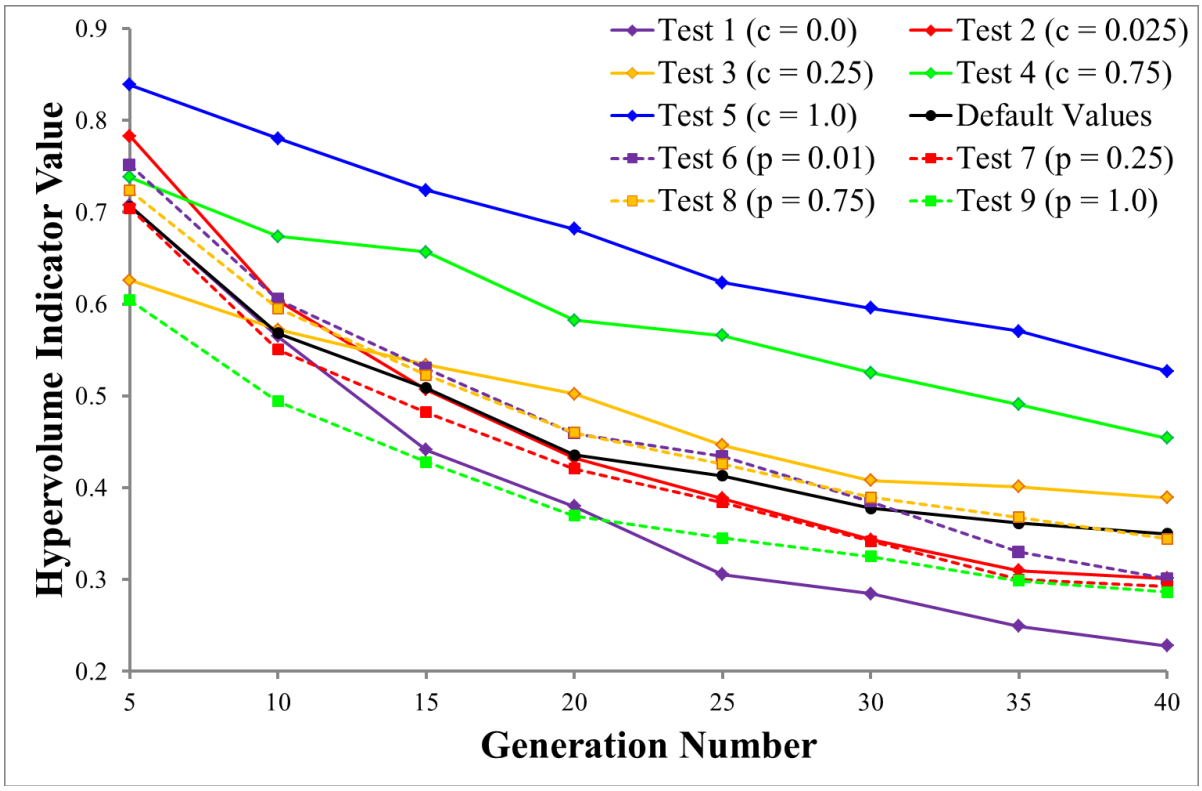

Fig. 13. The average hypervolume indicator value for each sensitivity study test every 5 generations.

Final hypervolume and epsilon indicator values were calculated (Table 10), as well as performing the Kruskal-Wallis test (Table 11) to determine whether observed differences are statistically significant. In this study, the Kruskal-Wallis test p-value results represent the probability that the difference between the performance indicator values for MOJADE runs with default parameter values and for each test could have occurred by chance. This is used to ascertain whether the performance of the DE algorithm MOJADE is significantly sensitive to changes to its control parameters.

Table 10
Sensitivity study hypervolume and epsilon indicator values: statistically significant results from the Kruskal-Wallis test (Table 11) are shown in **bold**.

Test Number	Hypervolume Indicator		Epsilon Indicator	
	Mean	Standard Deviation	Mean	Standard Deviation
Default values	0.3204	0.1324	3.0770	2.6952
1	0.2249	0.0798	**0.6934**	**0.8192**
2	0.2857	0.1065	1.1678	1.1824
3	0.3692	0.1489	2.0480	2.5493
4	**0.4319**	**0.1024**	1.7474	2.1725
5	**0.5146**	**0.0836**	4.1147	2.4837
6	0.2886	0.0619	**0.9181**	**0.9652**
7	0.2805	0.1064	**0.8389**	**0.8338**
8	0.3329	0.1145	2.2707	2.5475
9	0.2713	0.0956	1.2848	0.9661

Table 11
Sensitivity study Kruskal-Wallis test results: statistically significant results (< 0.05) are shown in **bold**.

Test	Hypervolume Indicator	Epsilon Indicator
1 vs default values	6.96E-02	**1.02E-02**
2 vs default values	9.40E-01	5.88E-02
3 vs default values	3.26E-01	1.12E-01
4 vs default values	**1.91E-02**	1.12E-01
5 vs default values	**5.20E-03**	4.06E-01
6 vs default values	8.80E-01	**2.84E-02**
7 vs default values	7.06E-01	**4.94E-02**
8 vs default values	5.45E-01	2.27E-01
9 vs default values	8.80E-01	1.99E-01

Table 11 shows that none of the tests show a statistically significant change in both performance indicators compared to algorithm performance with default values for parameter adaptation and greediness. This suggests that MOJADE is reasonably robust in handling nuclear fuel assembly design optimization problems with heterogeneous fuel types. There is some evidence to suggest that hypervolume performance does deteriorate at higher rates of parameter adaptation. The parameter adaptation rate controls the distributions from which mutation and crossover rates are chosen for new solutions. Successful (i.e. nondominated) solutions have their related crossover and mutation rates stored in normal and Cauchy distributions, respectively. Increasing the rate of parameter adaption increases the importance of crossover and mutation rates most recently added to the archive. Lower rates make the algorithm less adaptive as it searches, which could result in premature convergence and becoming trapped in local minima for highly non-linear problems. Excessively high rates, however, cause the algorithm to forget its 'history', and may end up performing the crossover and mutation operations too frequently (or not frequently enough). This can result in the creation of excessively random solutions, and thus slower convergence, or in the search becoming confined by existing solutions, which again leads to premature convergence.

There is also some evidence to suggest that epsilon indicator performance is affected by both parameter adaptation and greediness. Greediness directly impacts the diversity maintained in the population as the algorithm moves around the search space. An excessively greedy algorithm may not be able to maintain a sufficiently diverse population to properly explore the search space, whilst a lack of elitism can slow algorithm convergence. These results suggest that tuning of the MOJADE control parameters may produce results which are more consistently closer to the true Pareto front, but do not suggest that the true Pareto front itself can be improved further.

5. Conclusions

This study has introduced and investigated the use of multi-objective Differential Evolution algorithms for optimizing nuclear fuel assembly design problems. Beginning with a performance comparison against an EA on a typical problem, the multi-objective DE algorithms MOJADE and MOμJADE demonstrated that DE is able to find solutions comparable in quality to those found by MOAA and arguably better explore the search space of fuel pin patterns. Both DE algorithms exhibit good performance in this exploratory optimization problem, despite the algorithms originally being designed for single-objective optimization with a known global optimum.

MOJADE and MOμJADE were then tested on a more complex design problem involving both plutonium management and gadolinium distribution within a MOX assembly. Again the

DE algorithms were shown to be capable of generating designs that contain more plutonium compared to those from the reference literature and featured a lower PPF at BoL, with comparable performance over a 15 GWD/t cycle.

From these two problems it was concluded that MOJADE exhibits superior performance to MOμJADE. For the final test, the sensitivity of the performance of the MOJADE algorithm to the settings of its control parameters was investigated on the second problem. The two control parameters, the rate of parameter adaptation and the greediness of the algorithm, were varied and the relative performance of the algorithm was analyzed for statistical significance. The results indicate that MOJADE is robust to changes in its control parameters and does not require tuning to individual problems, which supports an earlier finding by Zhang and Sanderson (2009) on the underlying JADE algorithm.

This work demonstrates the effectiveness and reliability of DE algorithms as suitable multi-objective optimization metaheuristics for nuclear engineering design optimization problems. Further testing should tackle more complex problems with a wider range of objectives, including the introduction of thermal-hydraulic feedback mechanisms and the use of three-dimensional models for axial optimization of fuel zoning (e.g. for use in Boiling Water Reactors).

6. Data availability statement

To the best of the authors' knowledge, this paper and references herein contain all the data needed to reproduce and validate the results presented.

Acknowledgements

This work was produced as part of a PhD researching into the development of a multi-objective optimization capability for heterogeneous light water reactor fuel assemblies. The PhD is funded by the UK Engineering and Physical Sciences Research Council (EPSRC) through the Imperial College, University of Cambridge and Open University Centre for Doctoral Training in Nuclear Energy under grant EP/L015900/1ICO. The WIMS reactor physics software package produced by the ANSWERS Software Service at Wood Group plc (formerly Amec Foster Wheeler) and the assistance and expertise of the ANSWERS team have been very helpful during the course of this research.

References

Brown, C., Jin, Y., Leach, M., Hodgson, M., 2015. μJADE: adaptive differential evolution with a small population. Soft Comput. 1, 1–10.

Castillo, A., Ortiz-Servin, J.J., Perusquía, R., Silvestre, Y.C., 2011. Fuel lattice design with path relinking in BWR's. Prog. Nucl. Energy. 53 (4), 368–374.

Charles, A.J., Parks, G.T., 2017. Mixed Oxide LWR Assembly Design Optimization Using Differential Evolution Algorithms. 1–7 in Proc. ICONE25, Shanghai, PRC.

Coello Coello, C.A., Lamont, G.B., van Veldhuizen, D.A., 2007. Evolutionary Algorithms for Solving Multi-Objective Problems. 2nd edition. Springer, New York.

Deb, K., Pratap, A., Agarwal, S., Meyarivan, T., 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. 6 (2), 182–197.

Hirano, Y., Hida, K., Sakurada, K., Yamamoto, M., 1997. Optimization of fuel rod enrichment distribution to minimize rod power peaking throughout life with BWR fuel assembly. Nucl. Sci. Technol. 34 (1), 5–12.

Jagawa, S., Yoshii, T., Fukao, A., 2001. Boiling Water Reactor loading pattern optimization using simple linear perturbation and modified Tabu Search methods. Nucl. Sci. Eng. 138 (1), 67–77.

Khoshahval, F., Minuchehr, H., Zolfaghari, A., 2011. Performance evaluation of PSO and GA in PWR core loading pattern optimization. Nucl. Eng. Des. 241 (3), 799–808.

Knowles, J., Thiele, L., Zitzler, E., 2006. A Tutorial on the Performance Assessment of Stochastic Multiobjective Optimizers. TIK Report 214. Swiss Federal Institute of Technology (ETHZ), Zurich, Switzerland.

Kropaczek, D.J., Turinsky, P.J., 1991. In-core nuclear fuel management optimization for Pressurized Water Reactors utilizing Simulated Annealing. Nucl. Technol. 95, 9–32.

Kruskal, W.H., Wallis, W.A., 1952. Use of ranks in one-criterion variance analysis. J. Am. Stat. Assoc. 47 (260), 583–621.

Lattarulo, V., Lindley, B.A., Parks, G.T., 2014. Application of the MOAA for the optimization of CORAIL assemblies for nuclear reactors. 1413–1420 in Proc. IEEE Congress on Evolutionary Computation, Beijing, China.

Lattarulo, V., Parks, G.T., 2012. A preliminary study of a new multi-objective optimization algorithm. In Proc. IEEE Congress on Evolutionary Computation, Brisbane, Australia.

Lindley, B.A., et al., 2015. Release of WIMS10: a versatile reactor physics code for thermal and fast systems. 1793–1801 in Proc. ICAPP 2015, Nice, France.

Parks, G.T., 1996. Multiobjective Pressurized Water Reactor reload core design by nondominated Genetic Algorithm search. Nucl. Sci. Eng. 124 (1), 178–187.

Pereira, C.M., 2004. Evolutionary multicriteria optimization in core designs: basic investigations and case study. Ann. Nucl. Energy 31 (11), 1251–1264.

Sacco, W.F., Henderson, N., Rios-Coelho, A.C., Ali, M.M., Pereira, C.M.N.A., 2009. Differential evolution algorithms applied to nuclear reactor core design. Ann. Nucl. Energy 36 (8), 1093–1099.

Sawaragi, Y., Nakayama, H., Tanino, T., 1985. Theory of Multiobjective Optimization. Elsevier Science, 176 (1), 2–5.

Storn, R., Price, K., 1997. Differential Evolution – a simple and efficient heuristic for global optimization over continuous spaces. J. Global Optim. 11 (4), 341–359.

Suzuki, A., Kiyose, R., 1971. Application of Linear Programming to refueling optimization for light water moderated power reactors. Nucl. Sci. Eng. 46 (1),112–130.

Yamate, K., Mori, M., Ushio, T., Kawamura, M., 1997. Design of a gadolinia bearing mixed-oxide fuel assembly for Pressurized Water Reactors. Nucl. Eng. Des. 170 (1–3), 35–51.

Youinou, G., et al., 2001. Heterogeneous assembly for plutonium multi recycling in PWRs: The CORAIL concept. In Proc. GLOBAL 2001, Paris, France.

Zhang, J., and Sanderson, A.C., 2009. JADE: adaptive differential evolution with optional external archive. IEEE Trans. Evol. Comput. 13 (5), 945–958.

Zio, E., Viadana, G., 2011. Optimization of the inspection intervals of a safety system in a nuclear power plant by Multi-Objective Differential Evolution (MODE). Reliab. Eng. Syst. Safe. 96 (11) 1552–1563.

Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C.M., Grunert da Fonseca, V., 2003. Performance assessment of multiobjective optimizers: an analysis and review. IEEE Trans. Evol. Comput. 7 (2), 1–22.

Appendix A: Pseudocode of MOJADE and MOμJADE

Nomenclature

μCR = adaptive crossover probability
μF = adaptive mutation probability
A1 = archive used for dominated solutions
A2 = archive used for Pareto-equivalent solutions
BIR = restart variable used if no improvement is made
c = rate of parameter adaptation
D = number of dimensions (variables)
G = number of generations
meanA = arithmetic mean
meanL = Lehmer mean
NP = last member of the population
p = greediness of the mutation strategy
P = population
randn = normal distribution
randc = Cauchy distribution
SCR = set of successful crossover factors
SF = set of successful mutation factors
up_lim / low_lim = limits set by the variable bounds
b_i = crossover rate repair modifier following perturbation
v_i = ith test vector following mutation
u_i = ith test vector following crossover and perturbation
x_i = ith member of the population

MOJADE

```
Begin
   Set μCR = 0.5; μF = 0.5; A1, A2 = 0
   Create random initial population {x_{i,0} | i = 1, 2, …, NP}
   Evaluate and rank population, determine 100p% best vectors
   For g = 1 to G
      SF = 0, SCR = 0
      For i = 1 to NP
         CR_i = randn_i (μCR, 0.1), F_i = randc_i (μF, 0.1)
         Randomly choose x_{p_best} from 100p%
         Randomly choose x_{r1} =/= x_i from P
         Randomly choose x_{r2} =/= x_{r1} =/= x_i from P ∪ A1 + A2
         v_i = x_i + F_i · (x_{p_best} – x_i) + F_i · (x_{r1} – x_{r2})
         Generate jrand = randint(1, D)
         For j = 1 to D
            If j = jrand or rand(0, 1) < CR_i
               u_{i,j} = v_{i,j}
            Else
               u_{i,j} = x_{i,j}
            End If
```

```
        End For
        If f(u_i) dominates f(x_i)
            x_i → A1 (replaces random member of A1 if A1 is full)
            x_i = u_i
            CR_i → SCR, F_i → SF
        Else
            If f(u_i) is Pareto-equivalent to f(x_i)
            && f(u_i) is NOT dominated by f(A2)
                Remove members of A2 that are dominated by u_i
                u_i → A2
            End If
        End If
        Rerank 100p% best vectors
    End For
    μCR = (1 – c) · μCR + c · meanA(SCR)
    μF = (1 – c) · μF + c · meanL(SF)
End For
End
```

MO*μ*JADE

```
Begin
    Set μCR = 0.5; μF = 0.5; A1, A2 = 0
    Create random initial population {x_i,0|i = 1, 2, …, NP}
    Evaluate and rank population, determine 100p% best vectors
    For g = 1 to G
        SF = 0, SCR = 0
        For i = 1 to NP
            CR_i = randn_i (μCR, 0.1), F_i = randc_i (μF, 0.1)
            Randomly choose x_p_best from 100p%
            Randomly choose x_a =/= x_i =/= x_p_best from P
            Randomly choose x_b =/= x_a =/= x_i from P
            Randomly choose x_c from P ∪ A1 + A2
            v_i = x_i + F_i · (x_p_best – x_a) + F_i · (x_b – x_c)
            Generate jrand = randint(1, D)
            For j = 1 to D
                If j = jrand or rand(0, 1) < CR_i
                    u_i,j = v_i,j , b_i,j = 1
                Else
                    u_i,j = x_i,j , b_i,j = 0
                End If
            End For
            For j = 1 to D
                If rand(0, 1) ≤ 0.005
                    u_i,j = low_lim + rand(0, 1) · (up_lim – low_lim)
                    b_i,j = 0
                Else
                    u_i,j = u_i,j, b_i,j = b_i,j
                End If
            End For
```

```
        CRᵢ = ∑ b / D
        If f(uᵢ) dominates f(xᵢ)
            xᵢ → A1 (if A1 is full, replaces random member)
            xᵢ = uᵢ
            CRᵢ → SCR, Fᵢ → SF
        Else
            If f(uᵢ) is Pareto-equivalent to f(xᵢ)
            && f(uᵢ) is NOT dominated by f(A2)
                Remove members of A2 that uᵢ dominate
                uᵢ → A2
            End If
        End If
        Rerank 100p% best vectors
        If uᵢ ∪ 100p% best vectors
            BIR = BIR + 1
        End If
    End For
    If mod(g, max(100, 10D) = 0
        μCR = (1 – c) · μCR + c · meanA(SCR)
        μF = (1 – c) · μF + c · meanL(SF)
    End If
    If mod(g, max(1000, 100D) = 0
        If BIR== 0
            Reinitialize pop, include random member of 100p%
            BIR = 0
        End If
    End If
  End For
End
```